ROFL: Routing as the Firewall Layer

Hang Zhao
zhao@cs.columbia.edu
Columbia University

Chi-Kin Chau
chi-kin.chau@cl.cam.ac.uk
University of Cambridge

Steven M. Bellovin*
smb@cs.columbia.edu
Columbia University

Technical Report CUCS-026-08

Abstract

We propose a firewall architecture that treats port numbers as part of the IP address. Hosts permit connectivity to a service by advertising the IPaddr:port/48 address; they block connectivity by ensuring that there is no route to it. This design, which is especially well-suited to MANETs, provides greater protection against insider attacks than do conventional firewalls, but drops unwanted traffic far earlier than distributed firewalls do.

1 Introduction

Firewalls have long been a mainstay for enterprise network security [13]. However, changes in topology, connectivity, and use of the Internet has rendered them obsolescent at best. One alternative that has been proposed is the distributed firewall [3], where policy is enforced at each node.

Distributed firewalls have their own weakness, though: traffic flows all the way to the destination before being discarded. While this is not a problem from a host penetration perspective, it is problematic in bandwidth-constrained environments such as MANETs, or in the face of denial-of-service attacks. Accordingly, in [53] Zhao and Bellovin proposed a hybrid firewall, where nodes can outsource enforcement of their security policies. Hybrid firewalls pose a tradeoff between cost and risk: the further from the node security policies are enforced, the less transmission and filtering cost there; on the other hand, the risk of compromise is greater, since there are more nodes between a prospective target and the filtering point.

To solve this problem, we observe that firewalls are in some sense the dual of routing mechanisms. That is, the purpose of a firewall is to block certain traffic from flowing. By contrast, a routing mechanism defines how certain traffic should flow. We can use this to build a firewall: if there is no way for certain traffic to be routed towards a destination, the traffic is effectively blocked. We can block traffic at many points; the only constraint is that they should not know the route to the destination. Furthermore, we treat the port number as part of the IP address; this permits selective transmission to or block of specific services.

From this perspective, a hybrid firewall — one in which an exchange of messages controls the firewall function — corresponds to a routing protocol. That is, a routing protocol dynamically controls what paths traffic should take to a destination; a hybrid firewall dynamically controls what is blocked.

This paper describes ROFL: ROuting as the Firewall Layer.[1] We discuss the design of such a system, including both the advantages and disadvantages.

*Research was sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence and was accomplished under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the author(s) and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

[1]For reasons of taste, we forbear to include MANET Address Optimization in the title.

1.1 Structure of the Paper

Section 2 provides an overview of distributed and hybrid firewalls. Section 3 describes the ROFL scheme in considerably more detail. Potentially troublesome issues are covered in Section 4, along with an explanation of why we feel these are manageable.

One of the more interesting implications of using routing to implement firewalls is that we can draw on the large literature about routing to understand what will happen. One of the more interesting possibilities is mapping ROFL into the routing algebra framework [47, 25]. That will allow us to prove certain properties about ROFL firewalls, by drawing on the proofs already existing using the routing algebra. A complete description of that is beyond the scope of this paper; we do, however, include a brief sketch of how this might work.

We conclude by discussing related work and explaining why ROFL is particularly useful in MANETs.

2 Distributed and Hybrid Firewalls

Firewalls are effective means of protecting a local system or network of systems from network-based security threats. Conventional firewalls rely on the notion of restricted topology and control entry points to the network. All traffic from inside to outside, and vice versa, must pass through the firewall. Since traditional approach made assumptions on restricted network topology and trust on every single host in the Intranet, Bellovin [3] proposed a distributed solution to address the shortcomings. In the distributed firewall scheme, security policies are still centrally defined, but the enforcement is pushed to the individual end hosts. Security administrator defines security policy in terms of host identifiers. The resulting policy is then shipped out to each individual host that participates in the distributed firewall. The policy file is consulted when processing incoming or outgoing messages to verify their compliance.

The most obvious advantage of distributed firewall approach is that there is no longer a single choke point in terms of performance. Throughput is not limited by the speed of firewall processing. There is no longer a single point of failure between the inside and outside network. Araditional firewall does not have the knowledge of the intentions from each individual host. On the contrary, distributed firewall rely on the hosts to make appropriate and thus more secure decisions. Most importantly, distributed firewalls can protect hosts, such as those used by telecommuters, that are not within the topological boundary, regardless whether or not a tunnel has been set up.

Distributed firewalls have their own disadvantages: unwanted traffic flows all the way to the destination before being discarded. They introduce additional problem for end hosts with limited bandwidth supply and power capacity, such as mobile nodes in MANETs. Therefore, a hybrid solution is proposed in [53], which combines those two approaches together to achieve desired functionalities with lower cost. Security policy enforcement become quite flexible in hybrid firewall implementation. Policies can be enforced at end hosts in a distributed manner, or at some intermediate nodes which act as gateways protecting systems sitting behind.

A policy algebra was introduced to enable policy integration and delegation in hybrid firewall environments. It consists of five basic operations and enables many possible arrangements of security policies. For a quantitative study, cost and risk functions are introduced to associate with each policy enforcement. There are several attempts to define proper cost and risk metrics. For example, the number of rules in a packet filter rule set is a plausible cost metric, since additional CPU cycles are required to perform rule checking. On the other hand, the number of hops between the delegate and its protected host could be a reasonable risk metric since any single compromised host or link along the path between those two exposes security hole to the attacker. Hybrid firewalls pose a trade-off between cost and risk: the further from the node security policies are enforced, the less transmission and filtering cost there; on the other hand, the risk of compromise is greater.

Hybrid firewalls based on the policy algebra are able to perform fine-grained integration of security policies. They also facilitate the decentralization of security policy enforcement to adapt to the rapid topology changes in MANETs. More importantly, the algebra enables policy delegation in hybrid firewalls. Modern

heterogeneous networks consists of nodes with diversified energy supply and computation capacity. Hosts with lower computational power may outsource part of its rule set to a security delegate that it trusts, thus lowering its own cost. Unwanted traffic flows are then dropped early along the routing path to save transmission power, in particular for MANETs where energy conservation is crucial. Besides that, it reduces potential collisions among packet transmissions in MANETs, which often requires additional MAC layer efforts and collaborations to resolve. However, to balance the trade-off between cost and risk associated with policy enforcement, a global optimization algorithm is required to minimize overall cost for policy enforcement while maintaining the risk level below certain threshold. Such a centralized algorithm could be expensive and not scalable when the size of the network grows. To solve this problem, we treat firewalls as a form of routing and propose ROFL to develop a unifying approach of those two.

3 Firewalls as a Form of Routing

3.1 Routing to Ports

A conventional routing advertisement is of the form

$$R = \{p/m, M\}$$

where p is an address prefix, m is a prefix length, and M is a metric. We extend the prefix field to include a service s:

$$\{p : s/m, M\}$$

For example, to permit access to an SMTP server and a web server at 192.0.2.42, a node would originate the routes

$$\{192.0.2.42 : 25/48, M\}, \{192.0.2.42 : 80/48, M\}$$

where M is cost metric.

Port-specific routing advertisements are handled just like any other routing advertisements: the source node and any intermediate routers do a longest-prefix match on the advertisement. If there is no matching route, the packet is dropped.

This discussion glosses over two issues: IPv6[16] and protocols other than TCP [43]. IPv6 does not pose any special challenges. Extending, say, OSPF [38] to handle v6 was straight-forward [15]; the same would be true for ROFL, save that one would speak of /144s instead of /48s. We will not discuss this point any further.

Dealing with other protocols (i.e., UDP [42]) is more complex. Prepending or appending it to the port number interferes with aggregation points (Section 3.3). For now, we assume there is a separate routing table per protocol. (This can be viewed as prepending the protocol number to the IP address.) Another option would be to use the same number space for TCP and UDP port numbers, and ignoring collisions; that is only problematic if one wanted to permit access to, say, the TCP port, while blocking the UDP port. We defer further investigation of this point to future work.

Blocking a port is more complex. Suppose we wish to permit references to all ports on 192.0.2.42 *except* port 80. While we could advertise $2^{16} - 1$ routes $\{192.0.2.42 : 0/48, M\}, \{192.0.2.42 : 1/48, M\}, \{192.0.2.42 : 79/48, M\}, \{192.0.2.42 : 81/48, M\}, \ldots, \{192.0.2.42 : 65535/48, M\}$, this would be seen as an unfriendly thing to do to a routing table. (This is the fully decorrelated form [53]; clearly, one could advertise a shorter, correlated form, but it might still require a considerable number of prefix announcements.) Instead, we advertise an *infinity* route:

$$\{192.0.2.42 : 80/48, \infty\}$$

along with $\{192.0.2.42/32, M\}$. Infinity routes are entered into a router's *Forwarding Information Base* (FIB), but they are used as black hole (discard) entries. In effect, we treat the infinity route and the covering route to all services on the host as a correlated specification, compared with the fully decorrelated version given above [53, 32]. That said, the behavior of the two forms must be equivalent.

3.2 Client Routes

A service-specific route (that is, a route with a finite metric to a particular service) is suitable for use by servers. However, virtually all useful protocols require replies from servers to the clients. This in turn implies that there must be a route to client ports. There are several possible ways to handle this.

The first, and probably best, scheme is for client-like computers to advertise their non-server ports as service-specific routes. Current IANA standards[2] say that dynamic ports (i.e., those used for ordinary clients) should be in the range 49152-65535. For host h, then, all clients are in $h : 49152/34$; services are in $h : 32768/34$ or $h : 0/33$. (Note: 49,152 is $2^{15} + 2^{14}$, i.e., $1100\ 0000\ 0000\ 0000_2$.) Advertise an infinity route for those prefixes, to block access to any services that aren't specfically advertised via a /48. Then advertise a normal route to $h : 49152/34$ to permit access to all client ports on the machine. The few servers on higher-numbered ports can be covered by specific infinity routes. Alternatively, the full /48 can be announced, with specific infinity routes to block the few servers running.

A second scheme is for client machines to announce a /48 for client ports as they're used. This avoids the need for unused prefixes to exist when a client port is not in use. However, most machines are almost always running *something* that speaks on the network; besides, each routing announcement, even for a /48, will consume router CPU bandwidth, especially if the announcement reaches it via more than one path. Accordingly, this is probably not the best choice.

The third choice is a subtle but useful variant on the second: piggyback the routing announcement for the client's /48 on the initial SYN packet to the server. More precisely, send a special packet towards the server that contains both the /48 announcement and the initial packet for the server. Routers along the path from the client to the server install the appropriate reverse route, in effect nailing up a reverse circuit. However, it is not a circuit; if a router along this path were to fail, the normal routing protocol mechanisms would automatically reroute the traffic via some other path. Also note that this scheme would force all SYN packets to the router's slow path (that is, the CPU, rather than using the hardware routing on the line card); this might pose significant load. This is less of an issue for MANETs, where all routing is done by the CPU.

The latter two alternatives are particularly useful in environments that use Network Address Translators (NATs) [48, 26, 28]. NATs need to know when to create externally-visible mappings for internal ports. As such, they benefit from explicit announcements from clients about which port numbers are in use. (This is similar to what is done today with Universal Plug and Play (UPnP) devices [50].)

We note that today, most machines are either clients or servers; there are comparatively few machines that act as both. The exception is for things like the DNS and infrequent outbound administrative emails from servers. It may, then, be advisable to select one of the above mechanisms based on the normal role of the machine.

3.3 Aggregation Points

The obvious objection to ROFL is the growth of the routing table. We deal with this in more detail in Section 4; for now, though, we introduce the notion of *aggregation points*.

An aggregation point does what its name implies: it aggregates routes. More specifically, when sending out routing advertisements it discards all prefixes over a certain length, while ensuring that a relatively short prefix covers them all. In general, this is not a new concept. BGP contains specific mechanisms for it [44, Section 9.2.2.2]; even without those, the concept is an essential component of Classless Interdomain Routing (CIDR) [20]. Beyond that, some ISPs are known to filter (or philtre) too-long prefixes [45]. Generally speaking, prefix length filtering is done in places where the rest of the net does not need to know the details of internal topology.

With ROFL, the obvious place to put an aggregation point is at the border to the rest of the Internet. For MANETs, the gateway to the fixed network is the likely place. These are, of course, the usual locations for conventional firewalls. There is, however, a very important difference. With conventional firewalls, target

[2] `http://www.iana.org/assignments/port-numbers`

hosts are exposed to attack from any host on the inside. With ROFL, every router along the path from the attacker to the target acts as a firewall.

Consider, for example, a departmental server host h. It listens on port 25 to receive mail from the rest of the company; it also listens on port 80 for a departmental Wiki. That is, Wiki service is offered only within the department. Naturally, access from outside to both of these ports is blocked by the corporate firewall.

Suppose, though, that some off-LAN host has been subverted. With a conventional firewall, the Wiki port would be open to attack. With ROFL, however, only $h : 25/48$ is advertised; there is no route to the Wiki server. No router within the company will pass those packets. From the outside, where only a short prefix is known, attack packets wil flow towards the external-facing aggregation router; it, too, will drop them, much like a conventional firewall.

There is no need, however, to restrict aggregation points to border routers. Multilocation companies may do aggregation at each site boundary. Alternatively — and counterintuitively — the long, service-specific prefixes may be passed to other sites' gateway routers, but not readvertised within the site. This has the effect of dropping unwanted traffic before it traverses an expensive WAN link. We cannot do that with conventional firewalls, since it would imply trusting an external box. With ROFL, the outsourced filtering request is advisory; there are no security breakdown if the request is not heeded.

Another approach to aggregation is to view it as a tradeoff between routing table size on the one hand and security or traffic volume on the other. An aggregation point is a security filtering boundary; from beyond it, attack packets may flow towards the destination unhindered. By having fewer aggregation points, the filtering boundary is pushed further out; many more routers will participate in dropping unwanted traffic.

In principle, there is no need for a "one size fits all" policy towards aggregation points. Some services — for example, the main intranet web server — may need to be advertised very widely; conversely, other services (such as the aforementioned Wiki) may only be of interest to a few LANs. We call this a *policy region*. Doing this properly, though, requires configuration of muliple routers as well as the originating host. We leave the design of such a management system for future work.

4 Issues

4.1 Routing Table Size

As noted, ROFL will cause growth in local routing tables, and in particular in the FIB. There are three effects: total storage needed, time to handle routing updates, and time to look up a route. All depend on the number of prefixes; let us estimate the size of typical routing tables using ROFL.

The total number of routes that must be stored is the product of the number of hosts within an aggregation region and the average number of services per host. Table 1 shows the out-of-the-box configuration for some modern desktop operating systems; in addition, each host will consume one route for client ports, plus an infinity route to block server access. (If the host does not, in fact, run any servers, this latter can be omitted.) We do not consider server machines; in most environments, there are many fewer servers per LAN. It is clear that if very few servers are running, very few service prefixes will be announced for them. We thus assume that on average, each host will announce 13 prefixes.

The number of hosts within a region is much more variable, and of course depends on the size of the organization. That said, larger organizations will have larger routers. We can do some rough calculations based on RIR address allocation policies.

ARIN requires that organizations use about 80% of their assigned address space before they get more space. They hand out provider-independent address space to organizations that need at least a /20; i.e., to those who have more than about 3300 hosts. If we assume that the number of desktops is a bit larger than the number of employees, we will need to handle about $3300 \times 13 \approx 43,000$ prefixes. This is probably an upper bound; most (though of course not all) organizations with more employees than that are split across multiple locations, each of which is a separate aggregation region.

Table 1: The number of services running out-of-the-box on default installations of assorted desktop operating systems. The numbers are approximate, since different vendors will ship slightly different configurations. In addition, some ports (i.e., DHCP) are never used off-LAN, and hence would not occupy space in the enterprise routing table. In our tests, about half of the Vista ports were in the Vista client port range ($[2^{14} + 2^{15}, 2^{16} - 1]$); they may thus be covered by our client port scheme.

Operating System	*Services*
Ubuntu 7.10 desktop	3
Windows Vista	13
MacOS Leopard	2
NetBSD 4.0	0
FreeBSD 7.0	0
OpenBSD 4.2	3

43,000 prefixes is a lot; however, it is less than a fifth of the prefixes in a full Internet feed. Today's enterprise-grade routers are capable of handling this many prefixes. Estimates suggest that only 64M bytes of memory are needed, well below their design limits. It seems, then, that ROFL does not pose an unreasonable memory burden.

Routing table computation time may be more problematic. Let us first approach it analytically. OSPF is based on Dijkstra's Algorithm [17], which is $O(n^2)$ in the number of nodes. However, this bound is for relatively dense graphs. Many enterprise networks and most MANETs are much more sparsely connected; other algorithms can do much better in such cases. In particular, by suitable choice of data structures when implementing Dijkstra's algorithm, the running time can be reduced to $O(e \log_2 n)$ [30, 1].

The analytic approach tells us nothing about the constant, so it pays to look at real implementations as well. In [46], Shaikh and Greenberg calculated the CPU time needed for OSPF updates on a fully-connected, n-node topology, on a Cisco 7513 router. They do not say which model of Route Switch Processor was used; the time frame suggests that it was likely an RSP1 or RSP2. They found that the CPU time needed was approximately $.00000247n^2$ seconds. For 3,300 nodes, that would come to about 27 seconds, which is a bit on the high side. However, their paper was published in 2001. Furthermore, Cisco has a reputation for using relatively slow CPUs in their routers. Their current top-of-the-line model, the RSP16, has a 400 Mhz CPU; the RSP2 has a 100 Mhz CPU.[3] Accordingly, we suggest that we can scale that figure by a factor of at least 20, which brings the time to about 1.4 seconds, which is clearly acceptable.

There are more important reasons for optimism. Shaikh and Greenberg measured a fully-connected graph. Our topologies are *not* fully connected. There is little published data on enterprise networks; experience suggests, however, that most corporate locations outside of data centers have rather simple and often tree-like topologies. Let us be conservative and assume a fanout of 10 links per node. Using the formula given above, it will take about 2700 operations, compared with the 11,000,000 operations for the fully-connected graph. The CPU time needed should be almost unmeasurably low, even on old, slow CPUs.

There is an implicit assumption in this CPU time analysis that the number of prefixes is irrelevant. if there are no local policies that that cause some services to be routed differently than others, this will in fact be the case; the path between any two nodes will be the same regardless of the port number addressed. If that is not the case, let us approximate differently-routed prefixes as separate nodes. The total cost goes up proportionally to $\log_2$ of the number of nodes; let us continue our assumption that the number of links is 10 per node. We are thus dealing with about 43,000 and 4300 links; the result is only 25$\times$ larger than in the simpler case: still very small.

[3] `http://www.cisco.com/en/US/products/hw/routers/ps359/products_installation_and_configuration_guide_chapter09186a00801c63a5.html`

The remaining issue is lookup time. It does not appear to be an issue, either. [51] describes algorithms whose complexity is $O(\log_2 n)$ in prefix length. Doubling the prefix length from a nominal 24 bits on today's networks to 48 bits should cost only one extra memory reference. [7] describes a multilevel lookup algorithm; given the structure of our /48s, this will work extremely well, though their algorithm is not designed for rapid table updates.

From these analyses, we conclude that our scheme is quite feasible, probably on today's platforms and certainly with minor design changes.

4.2 Privacy and Scanning

A disadvantage of ROFL is that within the policy region, it exposes the addresses and services running on all internal nodes. This is indeed an issue; however, we feel that it is not serious.

Host-scanning is relatively easy. In enterprise networks, an on-LAN attacker can monitor ARP packets. (In a recent test by one of us, we detected almost 700 hosts on our department's network, simply by passively monitoring ARP packets.) In MANETs, most routing protocols disclose the identities of all nodes. Other host scanning strategies are described in [5].

Determining available services is not much harder, nor is it that much more intrusive. "Stealth scans"[4] are rarely noticed.

Passive techniques work, too. Eavesdropping on a MANET is straight-forward, given a compromised host. In an enterprise network, switches frequently "leak" packets intended for other ports; simply by sitting and listening, a host can learn of services run on many other machines. While ROFL makes a scanner's job somewhat easier, we do not feel that the difference is great enough to rule it out.

4.3 Policy Routing

Traditional firewalls have the ability to allow or block traffic based on source address as well as destination address and port number. While that is possible with ROFL, it is not as easy.

Doing route selection based in part on source addresses is a form of *policy routing*. Policy routing does not receive much attention today in the intradomain case; for an outline of an interdomain policy routing architecture, see [49, 18].

Using policy routing to allow traffic based on source addresses is of dubious utility; source-address spoofing is just too easy [37, 4, 31]. We noted in 1994 that "an attacker can still impersonate a host that is trusted by not on an internal network. One should not trust hosts outside of one's administrative control" [12, p. 67]. That said, it is reasonably straight-forward to encode source address restrictions as *constraints* attached to routes [40].

On the other hand, blocking certain source addresses is more useful, especially if the source of unwanted traffic is not address-agile. Again, this can be done with constraints on a path. The recent proposal to add such information to BGP [35] is intended for exactly that purpose:

> This mechanism is designed to, primarily, allow an upstream autonomous system to perform inbound filtering, in their ingress routers of traffic that a given downstream AS wishes to drop.

For most purposes, we believe that tagging ROFL routes with positive or negative constraints will suffice. If, however, local practic dictates that some services should be routed differently — for example, if traffic for VoIP services should be routed differently than bulk data transfers — it would be necessary to develop a full-blown policy routing protocol.

It should be noted that the security guarantees from any sort of source address constraints are not strong. As noted, source address spoofing can be used to evade permissive constraints; similarly, address agility — picking a new (and perhaps bogus) — address can be used to evade blocks. Furthermore, with ROFL the constraints are sent to each node within the policy region. Even if the transmission itself is encrypted,

[4] `http://www.insecure.org`

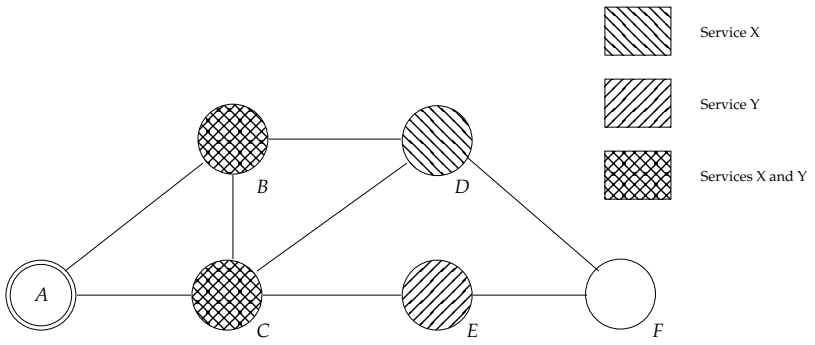

Figure 1: The desired scope of services announced from host A.

every internal node will be aware of every constraint, unless complex and expensive measures — one-way hashes of forbidden source addresses, to be checked on every packet? — are employed.

4.4 Defining Policy Regions

A crucial question in any firewall is defining which nodes should allowed access to the protected service. In ROFL, this translates to defining policy for which routing announcements are forwarded to which nodes.

It is straight-forward to cut off all services at the aggregation point. That works if the desired policy region is equivalent to all nodes behind that point. It does not help much if the desired scope for some service is less.

Consider the network shown in Figure 1, where node A is announcing services X and Y. Nodes B and C are allowed access to both services, while D is allowed access to X and E is allowed access to Y. F is not allowd access to either. In ROFL, this means that A should announce $\{A : X, A : Y\}$ to B and C; both of those nodes can forward $\{A : X\}$ to D, while E should see only $\{A : Y\}$ and F should see neither.

There are two obvious ways to approach this. First, all relevant nodes could have a policy definition statement. In C's case, for example, this statement would give the restrictions on what could be reannounced to D and E. This approach, though complex, might work for reasonably static nodes; it would not work well for MANETs. A better approach there would be to embed policy constraints in the routing announcements originating from A. In either case, the actual announcement becomes a form of routing policy; see Section 5 for more on that.

Regardless of which technique is used, the hard part is *understanding* the desired policy. This is primarily a human factors question.

4.5 Routing and Security

Since we are using to implement a security mechanism, it is fair to inquire if the underlying mechanism is secure. That is, if routing is insecure, are we building on network security on a foundation of sand.

It is well-known that routing protocols are insecure; see, for example, [41, 19, 4, 39, 33] and many more. However, to say this begs the question of what "insecure" means in this context.

Traditional routing security work has focused on the attacker trying to attract traffic, for examination, modification, or outright discard. That sort of misbehavior is not a threat to our scheme. Rather, we rely on routers not forwarding traffic for which they have no valid routes. For the most part, an attacker cannot influence this.

Consider a service announced within some policy region P. If the attacker is within P, there is no incremental danger from a routing attack; the attacker can already send evil packets towards the target.

Assume, then, that the attacker is not within P. Fraudulently announcing the service from there attracts packets to the attacker's node, but does nothing to send the packets towards the target.

The attacker's remaining choice is to compromise a node within P and send a fraudulent announcement to a neighboring router outside of P. This would indeed allow attack traffic to flow. However, the incremental risk is minimal. An attacker within P could just as easily launch the attack itself, or create a tunnel to the actual attacker. We thus conclude that there is no incremental risk of attack to using routing protocols as a firewall mechanism.

The situation is similar for infinity routes. An attacker might try to cause a denial of service attack by announcing fraudulent infinity routes to some node. However, the real announcement for that node will have a lower metric than ∞; as such, it will be preferred by uncorrupted routers.

The remaining threat is fraudulent announcement of a service by an enemy who for some reason cannot launch an attack directly. One possible scenario might be a man-in-the-middle attacker on some link who wished to remain covert. While we regard this as a low probability event, it can be dealt with via link security or by such mechanisms as [27], [21], or simple link-layer authentication.

5 Theory of Routing Algebra

Since ROFL is an architecture to deploy firewall through routing, the results of routing algebra will be useful to shed light on the robustness of firewalling by means of routing protocols. Routing algebra [47, 11] is a theoretical framework to capture and model general routing protocols, which have become increasingly more sophisticated and complex in recent years. In the past, routing protocols such as OSPF and RIP were usually designed to optimize a certain global metric (e.g. finding the shortest paths, or the most reliable paths). However, in the policy-driven settings as in inter-domain routing, different ISPs often have different considerations for selecting forwarding paths. For instance, BGP [44] allows a high degree of flexibility for customizing route import and export mechanisms at routers; this can create highly complex behaviour in inter-domain routing, driven by a variety of factors such as business relations, traffic engineering, and security concerns.

Unlike metric-optimizing routing protocols (e.g. OSFP and RIP), it has been reported in the literature that it is easy to construct settings of policy-based routing protocols with network-wide deleterious consequences. For instance, (1) there can be no consistent configuration to which the routing protocol can converge [36], or (2) there are non-unique and non-deterministic configurations given by the routing protocol [24, 10], or (3) even a consistent configuration exists, the routing prototocol still cannot converge to it under asynchronous communications [9, 10].

Sobrinho [47] proposes a generic routing algebra to understand the robust settings in policy-based routing, and derives a machine-checkable sufficient condition (called *monotonicity*) that guarantees the existence of consistent routing configuration. Griffin and Sobrinho [25] used routing algebra to develop a metalanguage (called metarouting) for constructing desirable settings that satisfies monotonicity.

Here we argue that the framework of routing algebra in [47, 11] is sufficiently generic to capture the mechanism of ROFL. By utilizing the same routing algebra to describe both routing and firewalling, we can develop a unifying approach for modeling, reasoning, and constructing robust routing and firewalling.

Routing algebra is motivated by the recognition that a path may not be only associated by a metric cost, but also can be abstractly associated by a signature. And the operation that translates a signature on an in-coming link to an out-going link can be heterogeneous on different nodes of the network.

The more general setting of routing algebra allows us to model sophisticated operations of ROFL, which otherwise can not be regarded as short-path routing in the Internet. For instance, the scenario of source-based policy regions in Section 4.4 can not be captured as a shortest path routing problem, which nonetheless can be subsumed by routing algebra.

We aim to prove that a sufficent condition that is similar to monotonicity in policy-based routing will be also useful to establish robust deployment of firewall by ROFL. We will use our policy algebra — a formal model of outsourced firewall policies — to map ROFL into the routing algebra. We will show how the deployment of firewalls can be checked in an automatic manner to guarantee that ROFL can converge

to a network-wide firewall enforcement in a deterministic manner, even in the presence of asynchronous communications among routers.

6 Related Work

The notion of using port numbers as part of the address is an old one. It was considered as part of the design of TCP/IP [8]. In the same time frame, Pup [6] and the Xerox Network System architecture [52] included the "socket number" as part of an address; the other two parts were a network number and a -bit host number. However, only the network number was used for routing.

Bellovin discussed including services as a component of IP addresses [2], as part of the process that led to the design of IPv6 [16]. Later, he and Gleitz proposed a design involving an IP address per process group [22], as an aid to firewalls.

A closer match to ROFL may be found in [35]. In it, a BGP [44] extension is proposed that describes a *flow specification*, i.e., a combination of a protocol number and the 4-tuple ⟨source host, source port, destination host, destination port⟩. The purpose of this scheme is allow early drop of traffic involved in a denial-of-service attack; as such, it can be used to implement Pushback [34, 29] or remotely-triggered black hole filtering (see, for example, [14] or [23]). As noted earlier, the authors did not intend it for positive action, i.e., routing *to* a service.

In a more philosophical vein, application firewalls [13] and Network Address Translators (NATs) [48, 26, 28] derive much of their protective power because the hosts behind them have addresses that are not advertised. In this sense, they rely on the same principle we are using: something that isn't addressable can't be reached.

7 Conclusions

We have presented a firewall mechanism that has significant advantages over conventional packet filters, pure distributed firewalls, and hybrid firewalls.

There are a number of extensions that may also prove useful. We mention them here; we may pursue them further after more analysis. One is to route on IPsec SPIs [32] instead of the port numbers. Without that, encrypted traffic can't take advantage of ROFL, since the port numbers are not visible to intermediate nodes. Routing on SPIs will permit early drop of many fake packets. It is especially useful if fine-grained SPIs are used, as in distributed firewalls.

A second extension would be to separate the port number from the IP address, and instead pass a list of permitted or prohibited port numbers along with a prefix announcement:

$$\{192.0.2.42/32, \{25, 80\}, M\}$$

That would permit a single announcment (and hence FIB entry) to handle many services. Perhaps more importantly, it would easily handle the case of an entire network wishing to permit a single service:

$$\{192.0.2.0/24, \{25, 80\}, M\}$$

On the other hand, this would require an additional lookup step not part of destination prefix matching; additionally, it might conflict with the desire to route different services differently.

Our scheme can be deployed in more or less any enterprise network. That said, it is especially well-suited to MANETs for several reasons:

- Edge-filtering in MANETs is less useful. Individual nodes are at much greater risk of physical compromise, especially in military scenarios. There is thus much more risk of an attack coming from within the firewall.

- MANETs tend to have a more complex internal connectivity graph. Traffic in a MANET tends to traverse many more router hops than in an enterprise network. In our design, this provides many more opportunities for filtering.
- MANET nodes are frequently battery-constrained, making forwarding a very expensive operation. Dropping unwanted traffic early is thus a major benefit, especially as compared with distributed firewalls.
- Many MANET routing protocols already carry full /32s for each node, since they cannot rely on on the topological properties of fixed networks that permit address aggregation. Furthermore, most MANET nodes run very few services — they rarely run Windows Vista! Consequently, the increment in the size of the routing table is quite modest, probably no more than a factor of 2 or 3.

ROFL does stretch the purposes of the routing system. That said, our analysis shows that implementations should be feasible, even on very modest equipment.

Acknowledgments

We would like to thank Randy Bush, Chris Morrow, Radia Perlman, and Philip Smith for much useful information and many useful comments.

References

[1] Alfred V. Aho, John E. Hopcroft, and Jeffrey D. Ullman. *The Design and Analysis of Computer Algorithms*. Addison-Wesley, 1974.

[2] S. Bellovin. On many addresses per host. RFC 1681, Internet Engineering Task Force, August 1994.

[3] Steven M. Bellovin. Distributed firewalls. *;login:*, pages 39–47, November 1999.

[4] Steven M. Bellovin. A look back at "Security problems in the TCP/IP protocol suite". In *Annual Computer Security Applications Conference*, December 2004. Invited paper.

[5] Steven M. Bellovin, Angelos Keromytis, and Bill Cheswick. Worm propagation strategies in an IPv6 Internet. *;login:*, pages 70–76, February 2006.

[6] David R. Boggs, John F. Shoch, Edward A. Taft, and Robert M. Metcalfe. Pup: An internetwork architecture. *IEEE Transactions on Communications*, COM-28(4):612–624, April 1980.

[7] Adam L. Buchsbaum, Glenn S. Fowler, Balachander Krishnamurthy, Kiem-Phong Vo, and Jia Wang. Fast prefix matching of bounded strings. In *Proceedings of ACM SIGACT ALENEX03*, Baltimore, Maryland, January 2003.

[8] Vint Cerf, 2004. Private conversation.

[9] Chi-Kin Chau. Policy-based Routing with Non-strict Preferences. In *Proc. ACM SIGCOMM*, September 2006.

[10] Chi-Kin Chau. A Game-theoretical Study of Robust Networked Systems. *To appear in IEEE Journal on Selected Areas in Communications, Special Issue of Game Theory in Communication Systems*, 2008.

[11] Chi-Kin Chau, Richard Gibbens, and Timothy G. Griffin. Towards a unifying theory for policy-based routing. In *Proc. IEEE INFOCOM*, April 2006.

[12] William R. Cheswick and Steven M. Bellovin. *Firewalls and Internet Security: Repelling the Wily Hacker*. Addison-Wesley, Reading, MA, first edition, 1994.

[13] William R. Cheswick, Steven M. Bellovin, and Aviel D. Rubin. *Firewalls and Internet Security; Repelling the Wily Hacker*. Addison-Wesley, Reading, MA, second edition, 2003.

[14] Remotely triggered black hole filter — destination based and source based. Cisco Systems, 2005. White paper.

[15] R. Coltun, D. Ferguson, and J. Moy. OSPF for IPv6. RFC 2740, Internet Engineering Task Force, December 1999.

[16] S. Deering and R. Hinden. Internet protocol, version 6 (IPv6) specification. RFC 2460, Internet Engineering Task Force, December 1998.

[17] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.

[18] Deborah Estrin and Martha Steenstrup. Inter domain policy routing: overview of architecture and protocols. *SIGCOMM Comput. Commun. Rev.*, 21(1):71–78, 1991.

[19] Gregory G. Finn. Reducing the vulnerability of dynamic computer networks. ISI Research Report ISI/RR-88-201, University of Southern California Information Sciences Institute, June 1988.

[20] V. Fuller and T. Li. Classless inter-domain routing (CIDR). RFC 4632, Internet Engineering Task Force, August 2006.

[21] V. Gill, J. Heasley, D. Meyer, P. Savola, and C. Pignataro. The generalized ttl security mechanism (GTSM). RFC 5082, Internet Engineering Task Force, October 2007.

[22] Peter M. Gleitz and Steven M. Bellovin. Transient addressing for related processes: Improved firewalling by using IPv6 and multiple addresses per host. In *Proceedings of the Eleventh Usenix Security Symposium,*, August 2001.

[23] Barry Raveendran Greene, Christopher L. Morrow, and Brian W. Gemberling. ISP security — real world techniques: Remote triggered black hole filtering and backscatter traceback. NANOG, October 2001.

[24] Timothy G. Griffin and Geoff Huston. RFC 4264: BGP wedgies, November 2005.

[25] Timothy G. Griffin and Joäo Luís Sobrinho. Metarouting. *SIGCOMM Comput. Commun. Rev.*, 35(4):1–12, September 2005.

[26] T. Hain. Architectural implications of NAT. RFC 2993, Internet Engineering Task Force, November 2000.

[27] A. Heffernan. Protection of BGP sessions via the TCP MD5 signature option. RFC 2385, Internet Engineering Task Force, August 1998.

[28] M. Holdrege and P. Srisuresh. Protocol complications with the IP network address translator. RFC 3027, Internet Engineering Task Force, January 2001.

[29] John Ioannidis and Steven M. Bellovin. Implementing pushback: Router-based defense against DDoS attacks. In *Proc. Internet Society Symposium on Network and Distributed System Security*, 2002.

[30] Donald B. Johnson. Efficient algorithms for shortest paths in sparse networks. *J. ACM*, 24(1):1–13, 1977.

[31] Laurent Joncheray. A simple active attack against TCP. In *Proceedings of the Fifth Usenix Unix Security Symposium*, Salt Lake City, UT, 1995.

[32] S. Kent and K. Seo. Security architecture for the Internet Protocol. RFC 4301, Internet Engineering Task Force, December 2005.

[33] Stephen Kent, Charles Lynn, and Karen Seo. Secure border gateway protocol (Secure-BGP). *IEEE Journal on Selected Areas in Communications*, 18(4):582–592, April 2000.

[34] Ratul Mahajan, Steven M. Bellovin, Sally Floyd, John Ioannidis, Vern Paxson, and Scott Shenker. Controlling high bandwidth aggregates in the network. *Computer Communications Review*, 32(3):62–73, July 2002.

[35] Pedro Marques, Nischal Sheth, Robert Raszuk, Barry Greene, Jared Mauch, and Danny McPherson. Dissemination of flow specification rules. Internet draft; work in progress, April 2008. (draft-ietf-idr-flow-spec-01.txt).

[36] D. McPherson, V. Gill, D. Walton, and A. Retana. RFC 3345: Border Gateway Protocol (BGP) persistent route oscillation condition, August 2002.

[37] Robert T. Morris. A weakness in the 4.2BSD unix TCP/IP software. Computing Science Technical Report 117, AT&T Bell Laboratories, Murray Hill, NJ, February 1985.

[38] J. Moy. OSPF version 2. RFC 2328, Internet Engineering Task Force, April 1998.

[39] S. Murphy, M. Badger, and B. Wellington. OSPF with digital signatures. RFC 2154, Internet Engineering Task Force, June 1997.

[40] Radia Perlman. Incorporation of service classes into a network architecture. *SIGCOMM Comput. Commun. Rev.*, 11(4):204–210, 1981.

[41] Radia Perlman. *Network Layer Protocols with Byzantine Robustness*. PhD thesis, M.I.T., 1988.

[42] J. Postel. User datagram protocol. RFC 768, Internet Engineering Task Force, August 1980.

[43] J. Postel. Transmission control protocol. RFC 793, Internet Engineering Task Force, September 1981.

[44] Y. Rekhter, T. Li, and S. Hares. A border gateway protocol 4 (BGP-4). RFC 4271, Internet Engineering Task Force, January 2006.

[45] Jennifer Rexford, Steven Bellovin, and Randy Bush. Some initial measurements of prefix length phyltreing. NANOG, May 2001.

[46] Aman Shaikh and Albert Greenberg. Experience in black-box ospf measurement. In *ACM SIGCOMM Internet Measurement Workshop (IMW)*, November 2001.

[47] João Luíz Sobrinho. An algebraic theory of dynamic network routing. *IEEE/ACM Trans. Networking*, 13(5):1160–1173, October 2005.

[48] P. Srisuresh and K. Egevang. Traditional IP network address translator (traditional NAT). RFC 3022, Internet Engineering Task Force, January 2001.

[49] M. Steenstrup. An architecture for Inter-Domain policy routing. RFC 1478, Internet Engineering Task Force, June 1993.

[50] WANIPConnection:1. Service Template Version 1.01, UPnP Forum, 12 November 2001. Standardized DCP.

[51] Marcel Waldvogel. *Fast Longest Prefix Matching: Algorithms, Analysis, and Applications*. PhD thesis, Swiss Federal Institute of Technology, Zurich, 2000.

[52] Xerox System Integration Standard. Internet transport protocols. XSIS 028112, Xerox Corporation, December 1981.

[53] Hang Zhao and Steven M. Bellovin. Policy algebras for hybrid firewalls. Technical Report CUCS-017-07, Department of Computer Science, Columbia University, March 2007. Also presented at the Annual Conference of the ITA, 2007.